Ex. 13

U.S. Patent No. 7,162,537 Infringement Chart

Title: Method and system for externally managing router configuration data in conjunction with a centralized database

Inventor: Pradeep Kathail

Abstract: A method and system for externally managing router configuration data in conjunction with a centralized database subsystem in a router device. The centralized database provides external management registration and unregistration for various managing router subsystems associated with said database system. The centralized database and the subsystems registered for external data management engage in transaction request sequences to provide router data requested by other client subsystems. The arrangement of the various client subsystems associated with the database subsystem allows the client subsystems to remain modular and independent of each other.

U.S. 7,162,537	Arista Products
[1.0] A method	To the extent the preamble is limiting, Arista switches, including at least the 7010, 7048, 7050, 7050X, 7150,
for reducing	7250X, 7280E, 7300, 7300X, and 7500E series models, and/or Arista EOS, including at least version 4.13.5F,
computational	perform a method for reducing computational overhead in a centralized database system by externally managing
overhead in a	router data in conjunction with a centralized database subsystem, said database subsystem operatively coupled for
centralized	communication with a plurality of router subsystems one of which is a first managing subsystem. The Arista 7150
database system	series, running EOS version 4.13.5F, is an exemplary model for demonstrating Arista's infringement of this patent.
by externally	
managing router	See e.g., Gary A. Donahue, Arista Warrior (2013) (App. M, Ex. 17) at p. 58 ("Arista switches have, at their heart, a
data in	database called SysDB. This database contains the state information and settings for the switch, organized in such a
conjunction	way that every module can access it with ease.").
with a	
centralized	See e.g., Gary A. Donahue, Arista Warrior (2013) (App. M, Ex. 17) at p. 58 ("Not only does EOS separate the
database	networking state from the processing, but drivers, processes, management, and even security patches run in user
subsystem, said	address space, not in the kernel.").
database	
subsystem	See e.g., Arista White Paper, EOS: The Next Generation Extensible Operating System (March 2014) (App. M, Ex.
operatively	14) at p. 4:
coupled for	
communication	

U.S. 7,162,537	Arista Products
with a plurality of router subsystems one of which is a first managing subsystem, comprising:	SysDB ASIC Driver SysDB ASIC Drivers ASIC STP OSPF Linux Kernel
	Figure 3: Arista EOS Architecture
	See e.g., Arista White Paper, EOS: The Next Generation Extensible Operating System (March 2014) (App. M, Ex. 14) at p. 5 ("As figure 3 shows, at the core of EOS is the system database. Sysdb is an in-memory database (machine generated at run time), which runs in user space and contains the complete state of the system. Like traditional databases, Sysdb does not contain any application logic and is only responsible for keeping state. However rather than being optimized for transactions, Sysdb is designed for synchronizing state among processes, also called 'agents', by notifying interested processes or agents when there is a change.").
	See, e.g., partial output from running "Show process" command on EOS v.4.13-5F:

J .S. 7,162,537			Ar	ista Products	
	localhost#show process				
			load average: 0.07, 0.22, 0.24		
	PID %CPU %MEM TT	STAT	STARTED TIME CMD		
	3191 2.0 0.0 pts/3				
	1606 5.6 4.1 ?	S	01:07:18 00:00:36 Sysdb	-d -idlopen -	
	1932 5.6 2.5 ?		01:07:46 00:00:34 FocalPoint	-d -idlopen -	
	1816 1.2 1.4 ?	S		-d -idlopen -	
	1608 1.0 2.8 ? 2058 0.9 1.9 ?	s s	01:07:18 00:00:06 Fru	-d -idlopen - -d -idlopen -	
	1860 0.8 2.7 ?	S	01:07:50 00:00:05 PhyAeluros 01:07:44 00:00:05 /usr/sbin/ribd		
	1607 0.7 2.2 ?	S	01:07:18 00:00:04 FastClid	-d -idlopen -	
	1609 0.7 2.1 ?	S	01:07:18 00:00:04 Launcher	-d -idlopen -	
	2408 0.6 1.9 pts/3		01:08:17 00:00:03 Cli [interac	-d -idlopen -	
	1777 0.5 1.8 ?	S	01:07:44 00:00:03 SuperServer	-d -idlopen -	
	1605 0.5 1.1 ?		01:07:18 00:00:03 ProcMgr-work	-d -idlopen -	
	1933 0.5 1.7 ?		01:07:46 00:00:03 Smbus	-d -idlopen -	
	1968 0.5 1.7 ?		01:07:47 00:00:03 Mdio	-d -idlopen -	
	1781 0.3 2.0 ?		01:07:44 00:00:02 Lag+LacpAgen	-d -idlopen -	
	1784 0.3 1.7 ?	S	01:07:44 00:00:01 Lldp	-d -idlopen -	
	2663 0.3 1.6 ?		01:08:23 00:00:01 Pmbus	-d -idlopen -	
	1990 0.2 1.4 ?	S	01:07:48 00:00:01 PhyEthtool	-d -idlopen -	
	1863 0.2 1.9 ?	S	01:07:44 00:00:01 IgmpSnooping	-d -idlopen -	
	1851 0.2 1.8 ?	S	01:07:44 00:00:01 Acl	-d -idlopen -	
	1786 0.2 1.8 ? 2041 0.1 1.8 ?	s s	01:07:44 00:00:01 LacpTxAgent 01:07:49 00:00:01 XcvrAgent	-d -idlopen - -d -idlopen -	
	1915 0.1 1.8 ?	S	01:07:49 00:00:01 X8VFAGENT	-d -idlopen -	
	1848 0.1 1.6 ?	S	01:07:44 00:00:01 Ebia	-d -idlopen -	
	1996 0.1 1.6 ?	S	01:07:48 00:00:00 PowerSupplyD	-d -idlopen -	
	1802 0.1 1.7 ?	S	01:07:44 00:00:00 Ira	-d -idlopen -	
	1 0.1 0.3 ?	Ss	01:06:00 00:00:01 /sbin/init		
	1792 0.1 1.6 ?	Sl	01:07:44 00:00:00 Aaa	-d -idlopen -	
	2038 0.1 1.6 ?		01:07:49 00:00:00 FanDetector	-d -idlopen -	
	1846 0.1 1.2 ?		01:07:44 00:00:00 Dot1x	-d -idlopen -	
	1899 0.1 1.7 ?		01:07:45 00:00:00 Ucd9012	-d -idlopen -	
	1864 0.1 1.6 ?	S	01:07:44 00:00:00 Thermostat	-d -idlopen -	
	1826 0.1 1.7 ?	S	01:07:44 00:00:00 Arp	-d -idlopen -	
	1805 0.1 1.7 ?	S	01:07:44 00:00:00 LedPolicy	-d -idlopen -	
	1913 0.1 1.6 ? 1885 0.1 1.5 ?	s s	01:07:46 00:00:00 ScdAgent	-d -idlopen -	
	1796 0.1 1.6 ?	S	01:07:45 00:00:00 Sb820 01:07:44 00:00:00 Mirroring	-d -idlopen - -d -idlopen -	
	1788 0.1 1.5 ?	S	01:07:44 00:00:00 Hilloling	-d -idlopen -	
	2400 0.1 0.3 ttyS0		01:08:16 00:00:00 FastCli	a i diopen	
	1088 0.1 0.0 ?	S	01:06:38 00:00:00 [kworker/0:2]		
	1855 0.1 1.2 ?		01:07:44 00:00:00 Fhrp	-d -idlopen -	
	1994 0.0 1.5 ?		01:07:48 00:00:00 Lm95234	-d -idlopen -	
	1998 0.0 1.5 ?		01:07:48 00:00:00 Lm73	-d -idlopen -	
	1797 0.0 1.2 ?		01:07:44 00:00:00 EventMon	-d -idlopen -	
	1862 0.0 1.6 ?	S	01:07:44 00:00:00 Qos	-d -idlopen -	
	1795 0.0 1.6 ?	S	01:07:44 00:00:00 PortSec	-d -idlopen -	
	1887 0.0 1.5 ?	S	01:07:45 00:00:00 PciBus	-d -idlopen -	
	1971 0.0 1.5 ?	S	01:07:47 00:00:00 Sol	-d -idlopen -	
	1868 0.0 1.5 ? 1836 0.0 1.2 ?	S	01:07:44 00:00:00 NetworkTopol 01:07:44 00:00:00 StpTopology	-d -idlopen - -d -idlopen -	
	1600 0.0 1.2 !	2	or.ou.st oo.oo.oo acbroborodă	-d -Idiopen -	

U.S. 7,162,537 **Arista Products** See, e.g., source code from \usr\\lib\python2.6\site-packages\Sysdbplugin\Stp.py in EOS v.4.13-5F at lns. 22-29: import SysdbPlugin.SysdbLauncher 23 agentName = 'Stp' 24 agentCfg = { 'name' : agentName, 25 'exe' : '/usr/bin/Stp', 26 'argv' : [], 'heartbeatPeriod' : 30 } 27 28 roleName = SysdbPlugin.SysdbLauncher.allSupervisorsRoleName SysdbPlugin.SysdbLauncher.agentConfigIs(entMan, roleName, agentCfg) 29 See e.g., Gary A. Donahue, Arista Warrior (2013) (App. M, Ex. 17) at p. 57 ("The operating system for Arista switches is called the Extensible Operating System, or EOS for short . . . Arista switches run Unix natively, but to make them easier for nonprogrammers to understand, EOS makes them look more like traditional (Cisco) networking devices."). See e.g., Arista At-A-Glance: Extensible Operating System (January 2010) (App. M, Ex. 4) at p. 2 ("Run what you want, and rest assured we have ample CPU capacity available with a dual-core 1.8Ghz AMD x86 CPU and trusted Linux scheduler."). See e.g., Arista White Paper, EOS: The Next Generation Extensible Operating System (March 2014) (App. M, Ex. 14) at p. 5 ("As figure 3 shows, at the core of EOS is the system database. Sysdb is an in-memory database (machine generated at run time), which runs in user space and contains the complete state of the system. Like traditional databases, Sysdb does not contain any application logic and is only responsible for keeping state. However rather than being optimized for transactions, Sysdb is designed for synchronizing state among processes, also called 'agents', by notifying interested processes or agents when there is a change. All agents in the system mount their configuration and status from Sysdb. This is very much like a file-system mount where read-only or read-write permissions are specified for each mount point. When an agent mounts from Sysdb, it receives its own local copy of all of the state in that mount point. As Sysdb is maintained in RAM, once the switch is turned off or restarted, information is lost. Sysdb is very much like an event-driven publish/subscribe model. If the state of an agent changes, Sysdb will send

Arista Products
an event notification to that agent, which will then update its local copy. Similarly when the agent writes to the mount point, it only changes its local copy and the write returns immediately.
Each EOS agent subscribes to Sysdb to be notified when the states of other agents change in Sysdb. When the state of an agent has been changed, this change notification is buffered and asynchronously sent to Sysdb, which then notifies all other agents who have subscribed to the changed agent.
This centralized database approach to passing state throughout the system, and the automated way the Sysdb code is generated reduces risk and error, improves software feature velocity, and provides flexibility for customers who can use the same APIs to receive notifications from Sysdb to customize and extend switch features.").
Arista switches, including at least the 7010, 7048, 7050, 7050X, 7150, 7250X, 7280E, 7300, 7300X, and 7500E series models, and/or Arista EOS, including at least version 4.13.5F, perform transmitting a management registration request by said first managing subsystem to said database subsystem, said registration request indicating router configuration data for which said first managing subsystem is requesting to provide external management services, said router configuration data managed by said database system and derived from configuration commands supplied by a user and executed by a router configuration subsystem before being stored in said database.
See e.g., Arista White Paper, EOS: The Next Generation Extensible Operating System (March 2014) (App. M, Ex. 14) at p. 5 ("As figure 3 shows, at the core of EOS is the system database. Sysdb is an in-memory database (machine generated at run time), which runs in user space and contains the complete state of the system. Like traditional databases, Sysdb does not contain any application logic and is only responsible for keeping state. However rather than being optimized for transactions, Sysdb is designed for synchronizing state among processes, also called 'agents', by notifying interested processes or agents when there is a change.
All agents in the system mount their configuration and status from Sysdb. This is very much like a file-system mount where read-only or read-write permissions are specified for each mount point. When an agent mounts from Sysdb, it receives its own local copy of all of the state in that mount point. As Sysdb is maintained in RAM, once the switch is turned off or restarted, information is lost. Sysdb is very much like an event-driven publish/subscribe model. If the state of an agent changes, Sysdb will send

U.S. 7,162,537	Arista Products
router	an event notification to that agent, which will then update its local copy. Similarly when the agent writes to the
configuration	mount point, it only changes its local copy and the write returns immediately.
data managed	
by said database	Each EOS agent subscribes to Sysdb to be notified when the states of other agents change in Sysdb When the state
system and	of an agent has been changed, this change notification is buffered and asynchronously sent to Sysdb, which then
derived from	notifies all other agents who have subscribed to the changed agent.").
configuration	
commands	See e.g., EOS: An Extensible Operating System (App. M, Ex. 15) at p. 3 ("Sysdb holds all state, while agents
supplied by a	perform all processing. Sysdb is an address space that purely holds state and delivers state updates from one agent
user and	to another. For example, when a link goes down, a port driver updates the link state in Sysdb, which delivers the
executed by a	update to the
router	SNMP service, which then sends a trap. Agents may contain copies of Sysdb state for efficiency, but all state is
configuration	recoverable from Sysdb whenever needed. Like a traditional database engine, Sysdb itself is ultra-reliable because
subsystem	it contains no application code.").
before being	
stored in said	See, e.g., source code from \usr\lib\python2.6\site-packages\Sysdbplugin\Stp.py in EOS v.4.13-5F at lns. 1-35:
database;	

```
U.S. 7,162,537
                                                             Arista Products
                  1
                       # Copyright (c) 2007-2011 Arista Networks, Inc. All rights reserved.
                  2
                       # Arista Networks, Inc. Confidential and Proprietary.
                  3
                       import Plugins
                  5
                       import Errdisable
                  6
                      @Plugins.plugin( requires=( 'interface/errdisable/causegroup',
                  8
                                                    'interface/errdisable/cause' ) )
                  9
                      def Plugin( entMan ):
                 10
                           # Create stp config status and hw.
                 11
                           entMan.register( 'stp/config', "Stp::Config" )
                 12
                           entMan.register( 'stp/portMode', "Tac::Dir" )
                 13
                           entMan.register( 'stp/input/config', "Tac::Dir" )
                 14
                           entMan.register( 'stp/input/config/cli', "Stp::Input::Config" )
                 15
                           entMan.register( 'stp/status', "Stp::Status" )
                           entMan.register( 'stp/protoStatus', "Stp::ProtoStatus" )
                 16
                 17
                           entMan.register( 'stp/hw', "Stp::Hw" )
                           entMan.register( 'stp/counter', "Stp::PortCounterDir" )
                 18
                           entMan.register( 'stp/standbyStatus', "Stp::StandbyStatus", force=True )
                 19
                 20
                           entMan.register( 'stp/ssoStableControlStatus', "Stp::StableControlStatus" )
                 21
                 22
                           import SysdbPlugin.SysdbLauncher
                 23
                           agentName = 'Stp'
                 24
                           agentCfg = { 'name' : agentName,
                 25
                                        'exe' : '/usr/bin/Stp',
                 26
                                        'argv' : [],
                 27
                                        'heartbeatPeriod' : 30 }
                 28
                           roleName = SysdbPlugin.SysdbLauncher.allSupervisorsRoleName
                 29
                           SysdbPlugin.SysdbLauncher.agentConfigIs( entMan, roleName, agentCfg )
                 30
                 31
                           entMan.registerLogFacility( 'SPANTREE' )
                 32
                           # create the bpduguard errdisable CauseStatus entity
                 33
                 34
                           causeDesc = "BPDU received on portfast port."
                           Errdisable.ErrdisableCauseGroupInit( entMan, 'bpduguard', False, causeDesc )
                 35
                See, e.g., Arista Configuration Guide v. 4.13.5F (4/2/14) (App. M, Ex. 25) at p. 898 ("Multiple Spanning Tree
```

Multiple Spanning Tree is enabled by the spanning-tree mode command STP version. Example This command enables Multiple Spanning Tree. switch (config) #spanning-tree mode mstp switch (config) #") (emphasis in original). See, e.g., Arista Configuration Guide v. 4.13.5F (4/2/14) (App. M, Ex. IDs for three MST instances:	25) at p. 90							
Example • This command enables Multiple Spanning Tree. switch (config) #spanning-tree mode mstp switch (config) #") (emphasis in original). See, e.g., Arista Configuration Guide v. 4.13.5F (4/2/14) (App. M, Ex. IDs for three MST instances:	25) at p. 90							
• This command enables Multiple Spanning Tree. switch (config) #spanning-tree mode mstp switch (config) #") (emphasis in original). See, e.g., Arista Configuration Guide v. 4.13.5F (4/2/14) (App. M, Ex. 1) IDs for three MST instances:	, -)2 ("TI	he switch defines bridge					
switch (config) # spanning-tree mode mstp switch (config) #'') (emphasis in original). See, e.g., Arista Configuration Guide v. 4.13.5F (4/2/14) (App. M, Ex. IDs for three MST instances:	, -)2 ("Tl	he switch defines bridge					
IDs for three MST instances:	, -)2 ("Tl	he switch defines bridge					
	01 5001 0							
\bullet MN D: 37/6x (Priority 137/6x)+Instance number(11) and D	016/301/2)3de (N	MAC address)					
	 MST 0: 32768 (Priority (32768)+Instance number(0)) and 001c.7301.23de (MAC address) MST101: 32869 (Priority (32768)+Instance number(101)) and 001c.7301.23de (MAC address) 							
	 MST101: 32809 (Friority (32768)+Instance number (101)) and 001c.7301.23de (MAC address) MST102: 32870 (Priority (32768)+Instance number (102)) and 001c.7301.23de (MAC address) 							
This command displays a table of root bridge information.	This command displays a table of root bridge information.							
switch>show spanning-tree root	switch>show spanning-tree root							
Root ID Root Hello M	lax Fv	wd						
Instance Priority MAC addr Cost Time A	ge D	ly	Root Port					
MST0 32768 001c.7301.23de 0 2	20		Po937					
	MST101 32869 001c.7301.23de 3998 0 0 0 Po909							
	MST102 32870 001c.7301.23de 3998 0 0 Po911							
(emphasis in original).								
indirectly. When you first log in to an Arista switch, the first thing you Since even your CLI session is a process with its own user space, the fi	See e.g., Gary A. Donahue, Arista Warrior (2013) (App. M, Ex. 17) at p. 58-59 ("You can see this in action, though indirectly. When you first log in to an Arista switch, the first thing you might do is issue the show run command. Since even your CLI session is a process with its own user space, the first time you issue the show run command, the process must mount the SysDB database. That takes a second or two, and you may notice the lag. After you get							

U.S. 7,162,537	Arista Products
	the output delivered, if you execute the show run command again, it delivers the output much faster because SysDB is already mounted. If you disconnect and then connect again, you'll spawn a new CLI process, which must then mount SysDB once more.").
	See e.g., Gary A. Donahue, Arista Warrior (2013) (App. M, Ex. 17) at p. 58-59 ("In fact, if you're impatient enough when first logging in and if you bang on the Enter key, you might be treated to the following message:
	Arista-7124SX login: admin waiting for mounts to complete ok Arista-7124SX>") (emphasis in original).
[1.2] b) receiving said management registration	Arista switches, including at least the 7010, 7048, 7050, 7050X, 7150, 7250X, 7280E, 7300, 7300X, and 7500E series models, and/or Arista EOS, including at least version 4.13.5F, perform receiving said management registration request by said database subsystem.
request by said database subsystem; and	See e.g., Arista White Paper, EOS: The Next Generation Extensible Operating System (March 2014) (App. M, Ex. 14) at p. 5 ("All agents in the system mount their configuration and status from Sysdb. This is very much like a file-system mount where read-only or read-write permissions are specified for each mount point. When an agent mounts from Sysdb, it receives its own local copy of all of the state in that mount point. As Sysdb is maintained in RAM, once the switch is turned off or restarted, information is lost.").
	See e.g., Arista White Paper, EOS: The Next Generation Extensible Operating System (March 2014) (App. M, Ex. 14) at p. 5 ("Each EOS agent subscribes to Sysdb to be notified when the states of other agents change in Sysdb. When the state of an agent has been changed, this change notification is buffered and asynchronously sent to Sysdb, which then notifies all other agents who have subscribed to the changed agent.").
	See, e.g., source code from \usr\lib\python2.6\site-packages\Sysdbplugin\Stp.py in EOS v.4.13-5F at lns. 22-29:

U.S. 7,162,537	Arista Products
	22 import SysdbPlugin.SysdbLauncher
	agentName = 'Stp'
	24 agentCfg = { 'name' : agentName,
	'exe': '/usr/bin/Stp',
	'argv': [],
	'heartbeatPeriod': 30 } roleName = SysdbPlugin.SysdbLauncher.allSupervisorsRoleName
	roleName = SysdbPlugin.SysdbLauncher.allSupervisorsRoleName SysdbPlugin.SysdbLauncher.agentConfigIs(entMan, roleName, agentCfg)
	Sysabriagin. Sysablauncher. agent configs (enthan, forewalke, agent cig)
[1.3] c) registering said first managing subsystem for external management by said database subsystem.	Arista switches, including at least the 7010, 7048, 7050, 7050X, 7150, 7250X, 7280E, 7300, 7300X, and 7500E series models, and/or Arista EOS, including at least version 4.13.5F, perform registering said first managing subsystem for external management by said managing subsystem. See e.g., Arista White Paper, EOS: The Next Generation Extensible Operating System (March 2014) (App. M, Ex. 14) at p. 5 ("All agents in the system mount their configuration and status from Sysdb. This is very much like a file-system mount where read-only or read-write permissions are specified for each mount point. When an agent mounts from Sysdb, it receives its own local copy of all of the state in that mount point. As Sysdb is maintained in RAM, once the switch is turned off or restarted, information is lost."). See e.g., Arista White Paper, EOS: The Next Generation Extensible Operating System (March 2014) (App. M, Ex. 14) at p. 5 ("Each EOS agent subscribes to Sysdb to be notified when the states of other agents change in Sysdb. When the state of an agent has been changed, this change notification is buffered and asynchronously sent to Sysdb, which then notifies all other agents who have subscribed to the changed agent."). See e.g., EOS: An Extensible Operating System (App. M, Ex. 15) at p. 3 ("Sysdb holds all state, while agents perform all processing. Sysdb is an address space that purely holds state and delivers state updates from one agent to another. For example, when a link goes down, a port driver updates the link state in Sysdb, which delivers the update to the SNMP service, which then sends a trap. Agents may contain copies of Sysdb state for efficiency, but all state is recoverable from Sysdb whenever needed. Like a traditional database engine, Sysdb itself is ultra-reliable because it contains no application code.").

U.S. 7,162,537	Arista Products
[10.0] A program storage	To the extent the preamble is limiting, Arista switches, including at least the 7010, 7048, 7050, 7050X, 7150, 7250X, 7280E, 7300, 7300X, and 7500E series models, and/or Arista EOS, including at least version 4.13.5F,
device readable	include a program storage device readable by a machine, tangibly embodying a program of instructions executable
by a machine,	by the machine to perform a method for reducing computational overhead in a centralized database system by
tangibly	externally managing router data in conjunction with a centralized database subsystem, said database subsystem
embodying a	operatively coupled for communication with a plurality of router subsystems. The Arista 7150 series, running EOS
program of	version 4.13.5F, is an exemplary model for demonstrating Arista's infringement of this patent.
instructions	
executable by	See e.g., Gary A. Donahue, Arista Warrior (2013) (App. M, Ex. 17) at p. 58 ("Arista switches have, at their heart, a
the machine to	database called SysDB. This database contains the state information and settings for the switch, organized in such a
perform a	way that every module can access it with ease.").
method for	
reducing	See e.g., Gary A. Donahue, Arista Warrior (2013) (App. M, Ex. 17) at p. 58 ("Not only does EOS separate the
computational	networking state from the processing, but drivers, processes, management, and even security patches run in user
overhead in a	address space, not in the kernel.").
centralized	Con and Ariota White Daner EOC: The Next Congretion Extensible Operating System (March 2014) (App. M. Ex
database system by externally	See e.g., Arista White Paper, EOS: The Next Generation Extensible Operating System (March 2014) (App. M, Ex. 14) at p. 4:
managing router	14) at p. 4.
data in	
conjunction	
with a	
centralized	
database	
subsystem, said	
database	
subsystem	
operatively	
coupled for	
communication	
with a plurality	

LED Driver SysDB ASIC Drivers ASIC Drivers ASIC Drivers Linux Kernel
See e.g., Arista White Paper, EOS: The Next Generation Extensible Operating System (March 2014) (App. M, Ex. 14) at p. 5 ("As figure 3 shows, at the core of EOS is the system database. Sysdb is an in-memory database (machine generated at run time), which runs in user space and contains the complete state of the system. Like traditional databases, Sysdb does not contain any application logic and is only responsible for keeping state. However rather than being optimized for transactions, Sysdb is designed for synchronizing state among processes, also called 'agents', by notifying interested processes or agents when there is a change."). See, e.g., output from running "Show process" command on EOS v.4.13-5F:

		Ar	rista Products	
localhost#show process				
		load average: 0.07, 0.22, 0.24		
PID %CPU %MEM TT		STARTED TIME CMD		
3191 2.0 0.0 pts/3 1606 5.6 4.1 ?	R+ S	01:17:59 00:00:00 ps -e -o pid,p 01:07:18 00:00:36 Sysdb	-d -idlopen -	
1932 5.6 2.5 ?		01:07:46 00:00:34 FocalPoint	-d -idlopen -	
1816 1.2 1.4 ?	S	01:07:44 00:00:07 AgentMonitor	-d -idlopen -	
1608 1.0 2.8 ?	S	01:07:18 00:00:06 Fru	-d -idlopen -	
2058 0.9 1.9 ?		01:07:50 00:00:05 PhyAeluros	-d -idlopen -	
1860 0.8 2.7 ?		01:07:44 00:00:05 /usr/sbin/ribd	-N	
1607 0.7 2.2 ?	S	01:07:18 00:00:04 FastClid	-d -idlopen -	
1609 0.7 2.1 ?	S	01:07:18 00:00:04 Launcher	-d -idlopen -	
2408 0.6 1.9 pts/3		01:08:17 00:00:03 Cli [interac	-d -idlopen -	
1777 0.5 1.8 ? 1605 0.5 1.1 ?	S	01:07:44 00:00:03 SuperServer 01:07:18 00:00:03 ProcMgr-work	-d -idlopen - -d -idlopen -	
1933 0.5 1.7 ?	5	01:07:46 00:00:03 Smbus	-d -idlopen -	
1968 0.5 1.7 ?	S	01:07:47 00:00:03 Mdio	-d -idlopen -	
1781 0.3 2.0 ?	S	01:07:44 00:00:02 Lag+LacpAgen	-d -idlopen -	
1784 0.3 1.7 ?	S	01:07:44 00:00:01 Lldp	-d -idlopen -	
2663 0.3 1.6 ?		01:08:23 00:00:01 Pmbus	-d -idlopen -	
1990 0.2 1.4 ?		01:07:48 00:00:01 PhyEthtool	-d -idlopen -	
1863 0.2 1.9 ?		01:07:44 00:00:01 IgmpSnooping	-d -idlopen -	
1851 0.2 1.8 ?	S	01:07:44 00:00:01 Acl	-d -idlopen -	
1786 0.2 1.8 ?	S	01:07:44 00:00:01 LacpTxAgent	-d -idlopen -	
2041 0.1 1.8 ? 1915 0.1 1.8 ?	S S	01:07:49 00:00:01 XcvrAgent 01:07:46 00:00:01 Ebra	-d -idlopen - -d -idlopen -	
1848 0.1 1.6 ?	3 S	01:07:44 00:00:01 Ebra	-d -idlopen -	
1996 0.1 1.6 ?	S	01:07:48 00:00:00 PowerSupplyD	-d -idlopen -	
1802 0.1 1.7 ?	S	01:07:44 00:00:00 Ira	-d -idlopen -	
1 0.1 0.3 ?	Ss	01:06:00 00:00:01 /sbin/init		
1792 0.1 1.6 ?	Sl	01:07:44 00:00:00 Aaa	-d -idlopen -	
2038 0.1 1.6 ?		01:07:49 00:00:00 FanDetector	-d -idlopen -	
1846 0.1 1.2 ?	S	01:07:44 00:00:00 Dot1x	-d -idlopen -	
1899 0.1 1.7 ?	S	01:07:45 00:00:00 Ucd9012	-d -idlopen -	
1864 0.1 1.6 ?	S	01:07:44 00:00:00 Thermostat	-d -idlopen -	
1826 0.1 1.7 ?	S	01:07:44 00:00:00 Arp	-d -idlopen -	
1805 0.1 1.7 ? 1913 0.1 1.6 ?	S S	01:07:44 00:00:00 LedPolicy 01:07:46 00:00:00 ScdAgent	-d -idlopen - -d -idlopen -	
1885 0.1 1.5 ?	S	01:07:45 00:00:00 Scaagent	-d -idlopen -	
1796 0.1 1.6 ?	S	01:07:44 00:00:00 Mirroring	-d -idlopen -	
1788 0.1 1.5 ?	S	01:07:44 00:00:00 Bfd	-d -idlopen -	
2400 0.1 0.3 ttyS0		01:08:16 00:00:00 FastCli		
1088 0.1 0.0 ?	S	01:06:38 00:00:00 [kworker/0:2]		
1855 0.1 1.2 ?		01:07:44 00:00:00 Fhrp	-d -idlopen -	
1994 0.0 1.5 ?		01:07:48 00:00:00 Lm95234	-d -idlopen -	
1998 0.0 1.5 ?		01:07:48 00:00:00 Lm73	-d -idlopen -	
1797 0.0 1.2 ?	S	01:07:44 00:00:00 EventMon	-d -idlopen -	
1862 0.0 1.6 ?	S	01:07:44 00:00:00 Qos	-d -idlopen -	
1795 0.0 1.6 ?	S	01:07:44 00:00:00 PortSec	-d -idlopen -	
1887 0.0 1.5 ? 1971 0.0 1.5 ?	S S	01:07:45 00:00:00 PciBus 01:07:47 00:00:00 Sol	-d -idlopen - -d -idlopen -	
1868 0.0 1.5 ?	S	01:07:47 00:00:00 SSI 01:07:44 00:00:00 NetworkTopol	-d -idlopen -	
1836 0.0 1.2 ?	S	01:07:44 00:00:00 StpTopology	-d -idlopen -	
		or o		

U.S. 7,162,537 **Arista Products** See, e.g., source code from \usr\\lib\python2.6\site-packages\Sysdbplugin\Stp.py in EOS v.4.13-5F at lns. 22-29: import SysdbPlugin.SysdbLauncher 23 agentName = 'Stp' 24 agentCfg = { 'name' : agentName, 25 'exe' : '/usr/bin/Stp', 26 'argv' : [], 27 'heartbeatPeriod' : 30 } 28 roleName = SysdbPlugin.SysdbLauncher.allSupervisorsRoleName SysdbPlugin.SysdbLauncher.agentConfigIs(entMan, roleName, agentCfg) 29 See e.g., Gary A. Donahue, Arista Warrior (2013) (App. M, Ex. 17) at p. 57 ("The operating system for Arista switches is called the Extensible Operating System, or EOS for short . . . Arista switches run Unix natively, but to make them easier for nonprogrammers to understand, EOS makes them look more like traditional (Cisco) networking devices."). See e.g., Arista At-A-Glance: Extensible Operating System (January 2010) (App. M, Ex. 4) at p. 2 ("Run what you want, and rest assured we have ample CPU capacity available with a dual-core 1.8Ghz AMD x86 CPU and trusted Linux scheduler."). See e.g., Arista White Paper, EOS: The Next Generation Extensible Operating System (March 2014) (App. M, Ex. 14) at p. 5 ("As figure 3 shows, at the core of EOS is the system database. Sysdb is an in-memory database (machine generated at run time), which runs in user space and contains the complete state of the system. Like traditional databases, Sysdb does not contain any application logic and is only responsible for keeping state. However rather than being optimized for transactions, Sysdb is designed for synchronizing state among processes, also called 'agents', by notifying interested processes or agents when there is a change. All agents in the system mount their configuration and status from Sysdb. This is very much like a file-system mount where read-only or read-write permissions are specified for each mount point. When an agent mounts from Sysdb, it receives its own local copy of all of the state in that mount point. As Sysdb is maintained in RAM, once the switch is turned off or restarted, information is lost. Sysdb is very much like an event-driven publish/subscribe model. If the state of an agent changes, Sysdb will send

U.S. 7,162,537	Arista Products
	an event notification to that agent, which will then update its local copy. Similarly when the agent writes to the mount point, it only changes its local copy and the write returns immediately.
	Each EOS agent subscribes to Sysdb to be notified when the states of other agents change in Sysdb. When the state of an agent has been changed, this change notification is buffered and asynchronously sent to Sysdb, which then notifies all other agents who have subscribed to the changed agent.
	This centralized database approach to passing state throughout the system, and the automated way the Sysdb code is generated reduces risk and error, improves software feature velocity, and provides flexibility for customers who can use the same APIs to receive notifications from Sysdb to customize and extend switch features.").
[10.1] (a) transmitting a management registration request by said first managing subsystem to said database	Arista switches, including at least the 7010, 7048, 7050, 7050X, 7150, 7250X, 7280E, 7300, 7300X, and 7500E series models, and/or Arista EOS, including at least version 4.13.5F, perform transmitting a management registration request by said first managing subsystem to said database subsystem, said registration request indicating router configuration data for which said first managing subsystem is requesting to provide external management services, said router configuration data managed by said database system and derived from configuration commands supplied by a user and executed by a router configuration subsystem before being stored in said database.
subsystem, said registration request indicating router configuration data for which said first	See e.g., Arista White Paper, EOS: The Next Generation Extensible Operating System (March 2014) (App. M, Ex. 14) at p. 5 ("As figure 3 shows, at the core of EOS is the system database. Sysdb is an in-memory database (machine generated at run time), which runs in user space and contains the complete state of the system. Like traditional databases, Sysdb does not contain any application logic and is only responsible for keeping state. However rather than being optimized for transactions, Sysdb is designed for synchronizing state among processes, also called 'agents', by notifying interested processes or agents when there is a change.
managing subsystem is requesting to provide external	All agents in the system mount their configuration and status from Sysdb. This is very much like a file-system mount where read-only or read-write permissions are specified for each mount point. When an agent mounts from Sysdb, it receives its own local copy of all of the state in that mount point. As Sysdb is maintained in RAM, once the switch is turned off or restarted, information is lost.
management services, said	Sysdb is very much like an event-driven publish/subscribe model. If the state of an agent changes, Sysdb will send

U.S. 7,162,537	Arista Products
router	an event notification to that agent, which will then update its local copy. Similarly when the agent writes to the
configuration	mount point, it only changes its local copy and the write returns immediately.
data managed	
by said database	Each EOS agent subscribes to Sysdb to be notified when the states of other agents change in Sysdb When the state
system and	of an agent has been changed, this change notification is buffered and asynchronously sent to Sysdb, which then
derived from	notifies all other agents who have subscribed to the changed agent.").
configuration	
commands	See e.g., EOS: An Extensible Operating System (App. M, Ex. 15) at p. 3 ("Sysdb holds all state, while agents
supplied by a	perform all processing. Sysdb is an address space that purely holds state and delivers state updates from one agent
user and	to another. For example, when a link goes down, a port driver updates the link state in Sysdb, which delivers the
executed by a	update to the SNMP service, which then sends a trap. Agents may contain copies of Sysdb state for efficiency, but
router	all state is recoverable from Sysdb whenever needed. Like a traditional database engine, Sysdb itself is ultra-
configuration	reliable because it contains no application code.").
subsystem	
before being	See, e.g., source code from \usr\lib\python2.6\site-packages\Sysdbplugin\Stp.py in EOS v.4.13-5F at lns. 1-35:
stored in said	
database;	

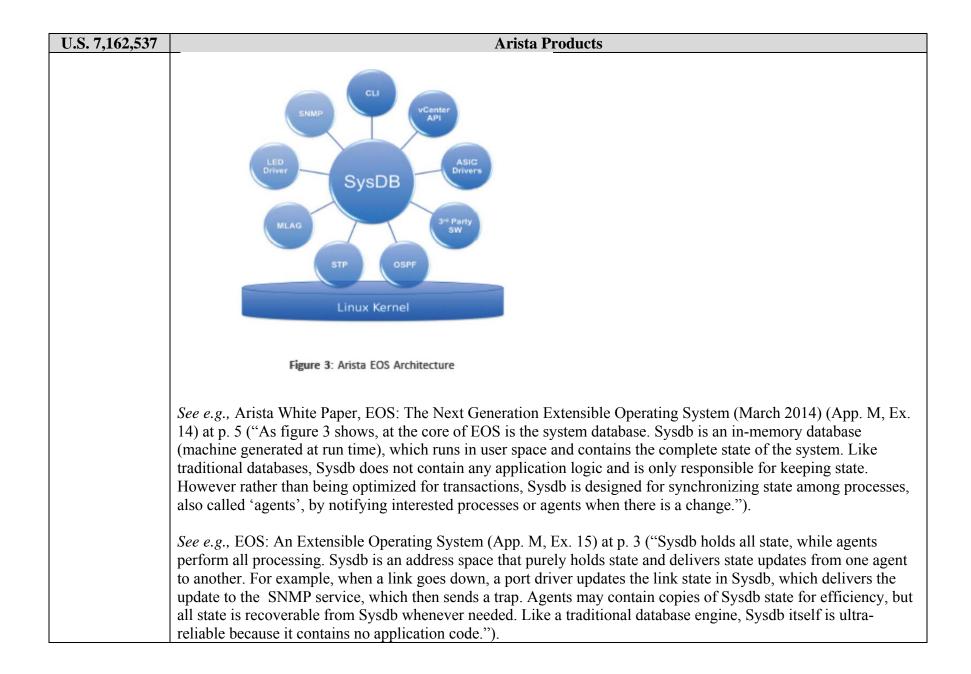
```
U.S. 7,162,537
                                                             Arista Products
                  1
                       # Copyright (c) 2007-2011 Arista Networks, Inc. All rights reserved.
                  2
                       # Arista Networks, Inc. Confidential and Proprietary.
                  3
                       import Plugins
                  5
                       import Errdisable
                  6
                      @Plugins.plugin( requires=( 'interface/errdisable/causegroup',
                  8
                                                    'interface/errdisable/cause' ) )
                  9
                      def Plugin( entMan ):
                 10
                           # Create stp config status and hw.
                 11
                           entMan.register( 'stp/config', "Stp::Config" )
                 12
                           entMan.register( 'stp/portMode', "Tac::Dir" )
                 13
                           entMan.register( 'stp/input/config', "Tac::Dir" )
                 14
                           entMan.register( 'stp/input/config/cli', "Stp::Input::Config" )
                 15
                           entMan.register( 'stp/status', "Stp::Status" )
                           entMan.register( 'stp/protoStatus', "Stp::ProtoStatus" )
                 16
                 17
                           entMan.register( 'stp/hw', "Stp::Hw" )
                           entMan.register( 'stp/counter', "Stp::PortCounterDir" )
                 18
                           entMan.register( 'stp/standbyStatus', "Stp::StandbyStatus", force=True )
                 19
                 20
                           entMan.register( 'stp/ssoStableControlStatus', "Stp::StableControlStatus" )
                 21
                 22
                           import SysdbPlugin.SysdbLauncher
                 23
                           agentName = 'Stp'
                 24
                           agentCfg = { 'name' : agentName,
                 25
                                        'exe' : '/usr/bin/Stp',
                 26
                                        'argv' : [],
                 27
                                        'heartbeatPeriod' : 30 }
                 28
                           roleName = SysdbPlugin.SysdbLauncher.allSupervisorsRoleName
                 29
                           SysdbPlugin.SysdbLauncher.agentConfigIs( entMan, roleName, agentCfg )
                 30
                 31
                           entMan.registerLogFacility( 'SPANTREE' )
                 32
                           # create the bpduguard errdisable CauseStatus entity
                 33
                 34
                           causeDesc = "BPDU received on portfast port."
                           Errdisable.ErrdisableCauseGroupInit( entMan, 'bpduguard', False, causeDesc )
                 35
                See, e.g., Arista Configuration Guide v. 4.13.5F (4/2/14) (App. M, Ex. 25) at p. 898 ("Multiple Spanning Tree
```

(MST) Multiple Spanning Tree is enabled by the spanning-tree mode command with the <i>mstp</i> option. MSTP is the defar STP version. Example This command enables Multiple Spanning Tree. switch (config) #spanning-tree mode mstp switch (config) #") (emphasis in original). See, e.g., Arista Configuration Guide v. 4.13.5F (4/2/14) (App. M, Ex. 25) at p. 902 ("The switch defines bridge IDs for three MST instances: MST 0: 32768 (Priority (32768)+Instance number(0)) and 001c.7301.23de (MAC address) MST 101: 32869 (Priority (32768)+Instance number(101)) and 001c.7301.23de (MAC address) MST 102: 32870 (Priority (32768)+Instance number(102)) and 001c.7301.23de (MAC address) This command displays a table of root bridge information. switch>show spanning-tree root Root ID Root Hello Max Fwd Instance Priority MAC addr Cost Time Age Dly Root Port MST 0 32768 001c.7301.23de 0 2 20 15 Po937 MST 101 32869 001c.7301.23de 3998 0 0 0 0 Po909 MST 102 32870 001c.7301.23de 3998 0 0 0 0 Po901 (emphasis in original).	U.S. 7,162,537			Arista	Products							
STP version. Example This command enables Multiple Spanning Tree. switch (config) #spanning-tree mode mstp switch (config) #") (emphasis in original). See, e.g., Arista Configuration Guide v. 4.13.5F (4/2/14) (App. M, Ex. 25) at p. 902 ("The switch defines bridge IDs for three MST instances: MST 0: 32768 (Priority (32768)+Instance number(0)) and 001c.7301.23de (MAC address) MST101: 32869 (Priority (32768)+Instance number(101)) and 001c.7301.23de (MAC address) MST102: 32870 (Priority (32768)+Instance number(102)) and 001c.7301.23de (MAC address) This command displays a table of root bridge information. Switch>show spanning-tree root Root ID Root Hello Max Fwd Instance Priority MAC addr Cost Time Age Dly Root Port MST0 32768 001c.7301.23de 0 2 20 15 Po937 MST101 32869 001c.7301.23de 3998 0 0 0 Po909 MST102 32870 001c.7301.23de 3998 0 0 0 Po909		(MST)										
This command enables Multiple Spanning Tree. switch (config) #spanning-tree mode mstp switch (config) #") (emphasis in original). See, e.g., Arista Configuration Guide v. 4.13.5F (4/2/14) (App. M, Ex. 25) at p. 902 ("The switch defines bridge IDs for three MST instances: • MST 0: 32768 (Priority (32768)+Instance number(0)) and 001c.7301.23de (MAC address) • MST101: 32869 (Priority (32768)+Instance number(101)) and 001c.7301.23de (MAC address) • MST102: 32870 (Priority (32768)+Instance number(102)) and 001c.7301.23de (MAC address) This command displays a table of root bridge information. switch>show spanning-tree root Root ID Root Hello Max Fwd Instance Priority MAC addr Cost Time Age Dly Root Port MST0 32768 001c.7301.23de 0 2 20 15 Po937 MST101 32869 001c.7301.23de 3998 0 0 0 Po909 MST102 32870 001c.7301.23de 3998 0 0 0 Po911		Multiple Spanning Tree is enabled by the spanning-tree mode command with the <i>mstp</i> option. MSTP is the default STP version.										
This command enables Multiple Spanning Tree. switch (config) #spanning-tree mode mstp switch (config) #") (emphasis in original). See, e.g., Arista Configuration Guide v. 4.13.5F (4/2/14) (App. M, Ex. 25) at p. 902 ("The switch defines bridge IDs for three MST instances: • MST 0: 32768 (Priority (32768)+Instance number(0)) and 001c.7301.23de (MAC address) • MST101: 32869 (Priority (32768)+Instance number(101)) and 001c.7301.23de (MAC address) • MST102: 32870 (Priority (32768)+Instance number(102)) and 001c.7301.23de (MAC address) This command displays a table of root bridge information. switch>show spanning-tree root Root ID Root Hello Max Fwd Instance Priority MAC addr Cost Time Age Dly Root Port MST0 32768 001c.7301.23de 0 2 20 15 Po937 MST101 32869 001c.7301.23de 3998 0 0 0 Po909 MST102 32870 001c.7301.23de 3998 0 0 0 Po911		Example										
switch (config) #") (emphasis in original). See, e.g., Arista Configuration Guide v. 4.13.5F (4/2/14) (App. M, Ex. 25) at p. 902 ("The switch defines bridge IDs for three MST instances: ■ MST 0: 32768 (Priority (32768)+Instance number(0)) and 001c.7301.23de (MAC address) ■ MST101: 32869 (Priority (32768)+Instance number(101)) and 001c.7301.23de (MAC address) ■ MST102: 32870 (Priority (32768)+Instance number(102)) and 001c.7301.23de (MAC address) This command displays a table of root bridge information. switch>show spanning-tree root Root ID Root Hello Max Fwd Instance Priority MAC addr Cost Time Age Dly Root Port MST0 32768 001c.7301.23de 0 2 20 15 Po937 MST101 32869 001c.7301.23de 3998 0 0 0 Po909 MST102 32870 001c.7301.23de 3998 0 0 0 Po901		•										
IDs for three MST instances: MST 0: 32768 (Priority (32768)+Instance number(0)) and 001c.7301.23de (MAC address) MST101: 32869 (Priority (32768)+Instance number(101)) and 001c.7301.23de (MAC address) MST102: 32870 (Priority (32768)+Instance number(102)) and 001c.7301.23de (MAC address) This command displays a table of root bridge information. Switch>show spanning-tree root												
 MST 0: 32768 (Priority (32768)+Instance number(0)) and 001c.7301.23de (MAC address) MST101: 32869 (Priority (32768)+Instance number(101)) and 001c.7301.23de (MAC address) MST102: 32870 (Priority (32768)+Instance number(102)) and 001c.7301.23de (MAC address) This command displays a table of root bridge information. switch>show spanning-tree root Root ID Root Hello Max Fwd Instance Priority MAC addr Cost Time Age Dly Root Port MST0 32768 001c.7301.23de 0 2 20 15 Po937 MST101 32869 001c.7301.23de 3998 0 0 0 Po909 MST102 32870 001c.7301.23de 3998 0 0 0 Po9011 												
MST101: 32869 (Priority (32768)+Instance number(101)) and 001c.7301.23de (MAC address) MST102: 32870 (Priority (32768)+Instance number(102)) and 001c.7301.23de (MAC address) This command displays a table of root bridge information. switch>show spanning-tree root Root ID				stance nur	nher((())) a:	nd 001c 73	801 23de	(MAC address)				
• MST102: 32870 (Priority (32768)+Instance number(102)) and 001c.7301.23de (MAC address) This command displays a table of root bridge information. switch>show spanning-tree root Root ID Root Hello Max Fwd Instance Priority MAC addr Cost Time Age Dly Root Port MST0 32768 001c.7301.23de 0 2 20 15 Po937 MST101 32869 001c.7301.23de 3998 0 0 0 Po909 MST102 32870 001c.7301.23de 3998 0 0 0 Po911												
switch>show spanning-tree root Root ID Root Hello Max Fwd Instance Priority MAC addr Cost Time Age Dly Root Port MST0 32768 001c.7301.23de 0 2 20 15 Po937 MST101 32869 001c.7301.23de 3998 0 0 0 Po909 MST102 32870 001c.7301.23de 3998 0 0 0 Po911												
Root ID Root Hello Max Fwd Fwd Root Fwd Fw		This command displays a table of root bridge information.										
Instance Priority MAC addr Cost Time Age Dly Root Port MST0 32768 001c.7301.23de 0 2 20 15 Po937 MST101 32869 001c.7301.23de 3998 0 0 0 Po909 MST102 32870 001c.7301.23de 3998 0 0 0 Po911		switch>show spanning-tree root										
Instance Priority MAC addr Cost Time Age Dly Root Port MST0 32768 001c.7301.23de 0 2 20 15 Po937 MST101 32869 001c.7301.23de 3998 0 0 0 Po909 MST102 32870 001c.7301.23de 3998 0 0 0 Po911			Root ID	Root	Hello	Max	Fwd					
MST101 32869 001c.7301.23de 3998 0 0 0 Po909 MST102 32870 001c.7301.23de 3998 0 0 0 Po911		Instance	Priority MAC addr		Time	Age	Dly	Root Port				
MST102 32870 001c.7301.23de 3998 0 0 0 Po911				_								
							_					
(Chiphasis in Original).				3998	0	0	0	Po911				
		(ciliphasis ili origin	ai).									
See e.g., Gary A. Donahue, Arista Warrior (2013) (App. M, Ex. 17) at p. 58-59 ("You can see this in action, thou indirectly. When you first log in to an Arista switch, the first thing you might do is issue the show run command. Since even your CLI session is a process with its own user space, the first time you issue the show run command the process must mount the SysDB database. That takes a second or two, and you may notice the lag. After you go		indirectly. When your CI	ou first log in to an Arista sv LI session is a process with i	witch, the fats own use	first thing er space, th	you might he first tim	do is issu e you issu	the show run command. ue the show run command,				

U.S. 7,162,537	Arista Products
	the output delivered, if you execute the show run command again, it delivers the output much faster because SysDB is already mounted. If you disconnect and then connect again, you'll spawn a new CLI process, which must then mount SysDB once more.").
	See e.g., Gary A. Donahue, Arista Warrior (2013) (App. M, Ex. 17) at p. 58-59 ("In fact, if you're impatient enough when first logging in and if you bang on the Enter key, you might be treated to the following message:
	Arista-7124SX login: admin waiting for mounts to complete ok Arista-7124SX>") (emphasis in original).
[10.2] (b) receiving said management registration	Arista switches, including at least the 7010, 7048, 7050, 7050X, 7150, 7250X, 7280E, 7300, 7300X, and 7500E series models, and/or Arista EOS, including at least version 4.13.5F, perform receiving said management registration request by said database subsystem.
request by said database subsystem; and	See e.g., Arista White Paper, EOS: The Next Generation Extensible Operating System (March 2014) (App. M, Ex. 14) at p. 5 ("All agents in the system mount their configuration and status from Sysdb. This is very much like a file-system mount where read-only or read-write permissions are specified for each mount point. When an agent mounts from Sysdb, it receives its own local copy of all of the state in that mount point. As Sysdb is maintained in RAM, once the switch is turned off or restarted, information is lost.").
	See e.g., Arista White Paper, EOS: The Next Generation Extensible Operating System (March 2014) (App. M, Ex. 14) at p. 5 ("Each EOS agent subscribes to Sysdb to be notified when the states of other agents change in Sysdb. When the state of an agent has been changed, this change notification is buffered and asynchronously sent to Sysdb, which then notifies all other agents who have subscribed to the changed agent.").
	See, e.g., source code from \usr\lib\python2.6\site-packages\Sysdbplugin\Stp.py in EOS v.4.13-5F at lns. 22-29:

U.S. 7,162,537	Arista Products
	22 import SysdbPlugin.SysdbLauncher
	23 agentName = 'Stp'
	24 agentCfg = { 'name' : agentName,
	'exe': '/usr/bin/Stp',
	26 'argv': [],
	'heartbeatPeriod': 30 }
	roleName = SysdbPlugin.SysdbLauncher.allSupervisorsRoleName
	SysdbPlugin.SysdbLauncher.agentConfigIs(entMan, roleName, agentCfg)
[10.3] (c)	Arista switches, including at least the 7010, 7048, 7050, 7050X, 7150, 7250X, 7280E, 7300, 7300X, and 7500E
registering said	series models, and/or Arista EOS, including at least version 4.13.5F, perform registering said first managing
first managing	subsystem for external management by said managing subsystem.
subsystem for	
external	See e.g., Arista White Paper, EOS: The Next Generation Extensible Operating System (March 2014) (App. M, Ex.
management by	14) at p. 5 ("All agents in the system mount their configuration and status from Sysdb. This is very much like a file-
said managing	system mount where read-only or read-write permissions are specified for each mount point. When an agent
subsystem.	mounts from Sysdb, it receives its own local copy of all of the state in that mount point. As Sysdb is maintained in RAM, once the switch is turned off or restarted, information is lost.").
	KAWI, once the switch is turned our of restarted, information is lost.).
	See e.g., Arista White Paper, EOS: The Next Generation Extensible Operating System (March 2014) (App. M, Ex.
	14) at p. 5 ("Each EOS agent subscribes to Sysdb to be notified when the states of other agents change in Sysdb.
	When the state of an agent has been changed, this change notification is buffered and asynchronously sent to
	Sysdb, which then notifies all other agents who have subscribed to the changed agent.").
	systes, which then notifies an other agents who have subscribed to the changed agent.).
	See e.g., EOS: An Extensible Operating System (App. M, Ex. 15) at p. 3 ("Sysdb holds all state, while agents
	perform all processing. Sysdb is an address space that purely holds state and delivers state updates from one agent
	to another. For example, when a link goes down, a port driver updates the link state in Sysdb, which delivers the
	update to the SNMP service, which then sends a trap. Agents may contain copies of Sysdb state for efficiency, but
	all state is recoverable from Sysdb whenever needed. Like a traditional database engine, Sysdb itself is ultra-
	reliable because it contains no application code.").

U.S. 7,162,537	Arista Products							
[19.0] In a	To the extent the preamble is limiting, Arista switches, including at least the 7010, 7048, 7050, 7050X, 7150,							
router device	7250X, 7280E, 7300, 7300X, and 7500E series models, and/or Arista EOS, including at least version 4.13.5F,							
having a	include, in a router device having a processor and memory, a router operating system executing within said							
processor and	memory. The Arista 7150 series, running EOS version 4.13.5F, is an exemplary model for demonstrating Arista's							
memory, a	infringement of this patent.							
router operating								
system	See e.g. 7150 Series 1/10 GbE SFP Ultra Low Latency Switch: Data Sheet (2012) (App. M, Ex. 1) at p. 6:							
executing	CPU Dual-Core x86							
within said	System Memory 4 Gigabytes							
memory	Flash Storage Memory 2 Gigabytes							
comprising:								
[19.1] (a) a	Arista switches, including at least the 7010, 7048, 7050, 7050X, 7150, 7250X, 7280E, 7300, 7300X, and 7500E							
database	series models, and/or Arista EOS, including at least version 4.13.5F, include a database subsystem.							
subsystem;								
	See e.g., Gary A. Donahue, Arista Warrior (2013) (App. M, Ex. 17) at p. 58 ("Arista switches have, at their heart, a							
	database called SysDB. This database contains the state information and settings for the switch, organized in such a							
	way that every module can access it with ease.").							
	See e.g., Gary A. Donahue, Arista Warrior (2013) (App. M, Ex. 17) at p. 58 ("Not only does EOS separate the							
	networking state from the processing, but drivers, processes, management, and even security patches run in user							
	address space, not in the kernel.").							
	See e.g., Arista White Paper, EOS: The Next Generation Extensible Operating System (March 2014) (App. M, Ex.							
	14) at p. 4:							



Case 5:14-cv-05344-BLF Document 306-13 Filed 06/13/16 Page 24 of 34

U.S. 7,162,537	Arista Products
	See, e.g., output from running "Show process" command on EOS v.4.13-5F:

		Ar	rista Products	
localhost#show process				
		load average: 0.07, 0.22, 0.24		
PID %CPU %MEM TT		STARTED TIME CMD		
3191 2.0 0.0 pts/3 1606 5.6 4.1 ?	R+ S	01:17:59 00:00:00 ps -e -o pid,p 01:07:18 00:00:36 Sysdb	-d -idlopen -	
1932 5.6 2.5 ?		01:07:46 00:00:34 FocalPoint	-d -idlopen -	
1816 1.2 1.4 ?	S	01:07:44 00:00:07 AgentMonitor	-d -idlopen -	
1608 1.0 2.8 ?	S	01:07:18 00:00:06 Fru	-d -idlopen -	
2058 0.9 1.9 ?		01:07:50 00:00:05 PhyAeluros	-d -idlopen -	
1860 0.8 2.7 ?		01:07:44 00:00:05 /usr/sbin/ribd	-N	
1607 0.7 2.2 ?	S	01:07:18 00:00:04 FastClid	-d -idlopen -	
1609 0.7 2.1 ?	S	01:07:18 00:00:04 Launcher	-d -idlopen -	
2408 0.6 1.9 pts/3		01:08:17 00:00:03 Cli [interac	-d -idlopen -	
1777 0.5 1.8 ? 1605 0.5 1.1 ?	S	01:07:44 00:00:03 SuperServer 01:07:18 00:00:03 ProcMgr-work	-d -idlopen - -d -idlopen -	
1933 0.5 1.7 ?	5	01:07:46 00:00:03 Smbus	-d -idlopen -	
1968 0.5 1.7 ?	S	01:07:47 00:00:03 Mdio	-d -idlopen -	
1781 0.3 2.0 ?	S	01:07:44 00:00:02 Lag+LacpAgen	-d -idlopen -	
1784 0.3 1.7 ?	S	01:07:44 00:00:01 Lldp	-d -idlopen -	
2663 0.3 1.6 ?		01:08:23 00:00:01 Pmbus	-d -idlopen -	
1990 0.2 1.4 ?		01:07:48 00:00:01 PhyEthtool	-d -idlopen -	
1863 0.2 1.9 ?		01:07:44 00:00:01 IgmpSnooping	-d -idlopen -	
1851 0.2 1.8 ?	S	01:07:44 00:00:01 Acl	-d -idlopen -	
1786 0.2 1.8 ?	S	01:07:44 00:00:01 LacpTxAgent	-d -idlopen -	
2041 0.1 1.8 ? 1915 0.1 1.8 ?	S S	01:07:49 00:00:01 XcvrAgent 01:07:46 00:00:01 Ebra	-d -idlopen - -d -idlopen -	
1848 0.1 1.6 ?	3 S	01:07:44 00:00:01 Ebra	-d -idlopen -	
1996 0.1 1.6 ?	S	01:07:48 00:00:00 PowerSupplyD	-d -idlopen -	
1802 0.1 1.7 ?	S	01:07:44 00:00:00 Ira	-d -idlopen -	
1 0.1 0.3 ?	Ss	01:06:00 00:00:01 /sbin/init		
1792 0.1 1.6 ?	Sl	01:07:44 00:00:00 Aaa	-d -idlopen -	
2038 0.1 1.6 ?		01:07:49 00:00:00 FanDetector	-d -idlopen -	
1846 0.1 1.2 ?	S	01:07:44 00:00:00 Dot1x	-d -idlopen -	
1899 0.1 1.7 ?	S	01:07:45 00:00:00 Ucd9012	-d -idlopen -	
1864 0.1 1.6 ?	S	01:07:44 00:00:00 Thermostat	-d -idlopen -	
1826 0.1 1.7 ?	S	01:07:44 00:00:00 Arp	-d -idlopen -	
1805 0.1 1.7 ? 1913 0.1 1.6 ?	S S	01:07:44 00:00:00 LedPolicy 01:07:46 00:00:00 ScdAgent	-d -idlopen - -d -idlopen -	
1885 0.1 1.5 ?	S	01:07:45 00:00:00 Scaagent	-d -idlopen -	
1796 0.1 1.6 ?	S	01:07:44 00:00:00 Mirroring	-d -idlopen -	
1788 0.1 1.5 ?	S	01:07:44 00:00:00 Bfd	-d -idlopen -	
2400 0.1 0.3 ttyS0		01:08:16 00:00:00 FastCli		
1088 0.1 0.0 ?	S	01:06:38 00:00:00 [kworker/0:2]		
1855 0.1 1.2 ?		01:07:44 00:00:00 Fhrp	-d -idlopen -	
1994 0.0 1.5 ?		01:07:48 00:00:00 Lm95234	-d -idlopen -	
1998 0.0 1.5 ?		01:07:48 00:00:00 Lm73	-d -idlopen -	
1797 0.0 1.2 ?	S	01:07:44 00:00:00 EventMon	-d -idlopen -	
1862 0.0 1.6 ?	S	01:07:44 00:00:00 Qos	-d -idlopen -	
1795 0.0 1.6 ?	S	01:07:44 00:00:00 PortSec	-d -idlopen -	
1887 0.0 1.5 ? 1971 0.0 1.5 ?	S S	01:07:45 00:00:00 PciBus 01:07:47 00:00:00 Sol	-d -idlopen - -d -idlopen -	
1868 0.0 1.5 ?	S	01:07:47 00:00:00 SSI 01:07:44 00:00:00 NetworkTopol	-d -idlopen -	
1836 0.0 1.2 ?	S	01:07:44 00:00:00 StpTopology	-d -idlopen -	
		or o		

U.S. 7,162,537	Arista Products
[19.2] (b) a plurality of client subsystems, each operatively coupled for communication to said database subsystem, one of said client subsystems configured as a managing subsystem to externally manage router data upon issuing a management request to said database subsystem; and	Arista switches, including at least the 7010, 7048, 7050, 7050X, 7150, 7250X, 7280E, 7300, 7300X, and 7500E series models, and/or Arista EOS, including at least version 4.13.5F, include a plurality of client subsystems, each operatively coupled for communication to said database subsystem, one of said client subsystems configured as a managing subsystem to externally manage router data upon issuing a management request to said database subsystem. See e.g., Arista White Paper, EOS: The Next Generation Extensible Operating System (March 2014) (App. M, Ex. 14) at p. 4:
	Figure 3: Arista EOS Architecture
	See e.g., Arista White Paper, EOS: The Next Generation Extensible Operating System (March 2014) (App. M, Ex. 14) at p. 5 ("As figure 3 shows, at the core of EOS is the system database. Sysdb is an in-memory database (machine generated at run time), which runs in user space and contains the complete state of the system. Like traditional databases, Sysdb does not contain any application logic and is only responsible for keeping state.

U.S. 7,162,537	Arista Products
	However rather than being optimized for transactions, Sysdb is designed for synchronizing state among processes, also called 'agents', by notifying interested processes or agents when there is a change.").
	See e.g., Arista White Paper, EOS: The Next Generation Extensible Operating System (March 2014) (App. M, Ex. 14) at p. 5 ("As figure 3 shows, at the core of EOS is the system database. Sysdb is an in-memory database (machine generated at run time), which runs in user space and contains the complete state of the system. Like traditional databases, Sysdb does not contain any application logic and is only responsible for keeping state. However rather than being optimized for transactions, Sysdb is designed for synchronizing state among processes, also called 'agents', by notifying interested processes or agents when there is a change.
	All agents in the system mount their configuration and status from Sysdb. This is very much like a file-system mount where read-only or read-write permissions are specified for each mount point. When an agent mounts from Sysdb, it receives its own local copy of all of the state in that mount point. As Sysdb is maintained in RAM, once the switch is turned off or restarted, information is lost.
	Sysdb is very much like an event-driven publish/subscribe model. If the state of an agent changes, Sysdb will send an event notification to that agent, which will then update its local copy. Similarly when the agent writes to the mount point, it only changes its local copy and the write returns immediately.
	Each EOS agent subscribes to Sysdb to be notified when the states of other agents change in Sysdb. When the state of an agent has been changed, this change notification is buffered and asynchronously sent to Sysdb, which then notifies all other agents who have subscribed to the changed agent.
	This centralized database approach to passing state throughout the system, and the automated way the Sysdb code is generated reduces risk and error, improves software feature velocity, and provides flexibility for customers who can use the same APIs to receive notifications from Sysdb to customize and extend switch features.").
[19.3] (c) a database operatively coupled to said database	Arista switches, including at least the 7010, 7048, 7050, 7050X, 7150, 7250X, 7280E, 7300, 7300X, and 7500E series models, and/or Arista EOS, including at least version 4.13.5F, include a database operatively coupled to said database subsystem, said database configured to store router configuration data and delegate management of router configuration data to a management subsystem that requests to manage router configuration data, said router configuration data managed by said database system and derived from configuration commands supplied by a user

U.S. 7,162,537	Arista Products
subsystem, said	and executed by a router configuration subsystem before being stored in said database.
database	
configured to	See e.g., Gary A. Donahue, Arista Warrior (2013) (App. M, Ex. 17) at p. 58 ("Arista switches have, at their heart, a
store router	database called SysDB. This database contains the state information and settings for the switch, organized in such a
configuration	way that every module can access it with ease.").
data and	
delegate	See e.g., Gary A. Donahue, Arista Warrior (2013) (App. M, Ex. 17) at p. 58 ("Not only does EOS separate the
management of	networking state from the processing, but drivers, processes, management, and even security patches run in user
router	address space, not in the kernel.").
configuration	
data to a	See e.g., Arista White Paper, EOS: The Next Generation Extensible Operating System (March 2014) (App. M, Ex.
management	14) at p. 4:
subsystem that	
requests to	
manage router	CLI
configuration	SNMP VCenter API
data, said router	
configuration	LED ASIC
data managed	SysDB Drivers
by said database	Gyobb
system and	3rd Party
derived from	MLAG 3** Party SW
configuration	
commands	STP OSPF
supplied by a	
user and	Linux Kernel
executed by a	
router	
configuration	Figure 3: Arista EOS Architecture
subsystem	
before being	
stored in said	See e.g., Arista White Paper, EOS: The Next Generation Extensible Operating System (March 2014) (App. M, Ex.

U.S. 7,162,537	Arista Products
database.	14) at p. 5 ("As figure 3 shows, at the core of EOS is the system database. Sysdb is an in-memory database (machine generated at run time), which runs in user space and contains the complete state of the system. Like traditional databases, Sysdb does not contain any application logic and is only responsible for keeping state. However rather than being optimized for transactions, Sysdb is designed for synchronizing state among processes, also called 'agents', by notifying interested processes or agents when there is a change.
	All agents in the system mount their configuration and status from Sysdb. This is very much like a file-system mount where read-only or read-write permissions are specified for each mount point. When an agent mounts from Sysdb, it receives its own local copy of all of the state in that mount point. As Sysdb is maintained in RAM, once the switch is turned off or restarted, information is lost.
	Sysdb is very much like an event-driven publish/subscribe model. If the state of an agent changes, Sysdb will send an event notification to that agent, which will then update its local copy. Similarly when the agent writes to the mount point, it only changes its local copy and the write returns immediately.
	Each EOS agent subscribes to Sysdb to be notified when the states of other agents change in Sysdb. When the state of an agent has been changed, this change notification is buffered and asynchronously sent to Sysdb, which then notifies all other agents who have subscribed to the changed agent.
	This centralized database approach to passing state throughout the system, and the automated way the Sysdb code is generated reduces risk and error, improves software feature velocity, and provides flexibility for customers who can use the same APIs to receive notifications from Sysdb to customize and extend switch features.").
	See e.g., EOS: An Extensible Operating System (App. M, Ex. 15) at p. 3 ("Sysdb holds all state, while agents perform all processing. Sysdb is an address space that purely holds state and delivers state updates from one agent to another. For example, when a link goes down, a port driver updates the link state in Sysdb, which delivers the update to the SNMP service, which then sends a trap. Agents may contain copies of Sysdb state for efficiency, but all state is recoverable from Sysdb whenever needed. Like a traditional database engine, Sysdb itself is ultrareliable because it contains no application code.").
	See e.g., EOS: An Extensible Operating System (App. M, Ex. 15) at p. 3 ("Sysdb holds all state, while agents perform all processing. Sysdb is an address space that purely holds state and delivers state updates from one agent

U.S. 7,162,537			Arista	Products							
	to another. For example, when a link goes down, a port driver updates the link state in Sysdb, which delivers the update to the SNMP service, which then sends a trap. Agents may contain copies of Sysdb state for efficiency, but all state is recoverable from Sysdb whenever needed. Like a traditional database engine, Sysdb itself is ultrareliable because it contains no application code.").										
	See, e.g., Arista Configuration Guide v. 4.13.5F (4/2/14) (App. M, Ex. 25) at p. 898 (" Multiple Spanning Tree (MST)										
	Multiple Spanning Tree is enabled by the spanning-tree mode command with the <i>mstp</i> option. MSTP is the default STP version.										
	Example										
	_	enables Multiple Spanning	Tree.								
	switch (config) # spanning-tree mode mstp switch (config) #") (emphasis in original).										
	See, e.g., Arista Configuration Guide v. 4.13.5F (4/2/14) (App. M, Ex. 25) at p. 902 ("The switch defines bridge IDs for three MST instances:										
	• MST 0: 32768 (Priority (32768)+Instance number(0)) and 001c.7301.23de (MAC address)										
		1: 32869 (Priority (32768)+		`	//		`	· · · · · · · · · · · · · · · · · · ·			
	• MST10	2: 32870 (Priority (32768)+	Instance	number(102	2)) and 001	c./301.2	3de (MA	C address)			
	This command displays a table of root bridge information.										
	switch>show spanning-tree root										
		Root ID	Root	Hello	Max	Fwd					
	Instance	Priority MAC addr	Cost	Time	Age	Dly		Root Port			
	MST0	32768 001c.7301.23de	0	2	20		Po937				
	MST101	32869 001c.7301.23de	3998	0	0	0	Po909				
	MST102	32870 001c.7301.23de	3998	0	0	0	Po911				

Case 5:14-cv-05344-BLF Document 306-13 Filed 06/13/16 Page 31 of 34

U.S. 7,162,537	Arista Products
	(emphasis in original).
	See, e.g., source code from \usr\lib\python2.6\site-packages\Sysdbplugin\Stp.py in EOS v.4.13-5F at lns. 1-35:

```
U.S. 7,162,537
                                                             Arista Products
                  1
                       # Copyright (c) 2007-2011 Arista Networks, Inc. All rights reserved.
                  2
                       # Arista Networks, Inc. Confidential and Proprietary.
                  3
                  4
                       import Plugins
                  5
                       import Errdisable
                  6
                      @Plugins.plugin( requires=( 'interface/errdisable/causegroup',
                  8
                                                    'interface/errdisable/cause' ) )
                  9
                      def Plugin( entMan ):
                 10
                           # Create stp config status and hw.
                 11
                           entMan.register( 'stp/config', "Stp::Config" )
                 12
                           entMan.register( 'stp/portMode', "Tac::Dir" )
                 13
                           entMan.register( 'stp/input/config', "Tac::Dir" )
                 14
                           entMan.register( 'stp/input/config/cli', "Stp::Input::Config" )
                 15
                           entMan.register( 'stp/status', "Stp::Status" )
                           entMan.register( 'stp/protoStatus', "Stp::ProtoStatus" )
                 16
                 17
                           entMan.register( 'stp/hw', "Stp::Hw" )
                           entMan.register( 'stp/counter', "Stp::PortCounterDir" )
                 18
                 19
                           entMan.register( 'stp/standbyStatus', "Stp::StandbyStatus", force=True )
                 20
                           entMan.register( 'stp/ssoStableControlStatus', "Stp::StableControlStatus" )
                 21
                 22
                           import SysdbPlugin.SysdbLauncher
                 23
                           agentName = 'Stp'
                 24
                           agentCfg = { 'name' : agentName,
                 25
                                        'exe' : '/usr/bin/Stp',
                 26
                                        'argv' : [],
                 27
                                       'heartbeatPeriod' : 30 }
                 28
                           roleName = SysdbPlugin.SysdbLauncher.allSupervisorsRoleName
                 29
                           SysdbPlugin.SysdbLauncher.agentConfigIs( entMan, roleName, agentCfg )
                 30
                 31
                           entMan.registerLogFacility( 'SPANTREE' )
                 32
                           # create the bpduguard errdisable CauseStatus entity
                 33
                 34
                           causeDesc = "BPDU received on portfast port."
                           Errdisable.ErrdisableCauseGroupInit( entMan, 'bpduguard', False, causeDesc )
                 35
                See e.g., Arista White Paper, EOS: The Next Generation Extensible Operating System (March 2014) (App. M, Ex.
```

U.S. 7,162,537	Arista Products
	14) at p. 5 ("As figure 3 shows, at the core of EOS is the system database. Sysdb is an in-memory database (machine generated at run time), which runs in user space and contains the complete state of the system. Like traditional databases, Sysdb does not contain any application logic and is only responsible for keeping state. However rather than being optimized for transactions, Sysdb is designed for synchronizing state among processes, also called 'agents', by notifying interested processes or agents when there is a change.
	All agents in the system mount their configuration and status from Sysdb. This is very much like a file-system mount where read-only or read-write permissions are specified for each mount point. When an agent mounts from Sysdb, it receives its own local copy of all of the state in that mount point. As Sysdb is maintained in RAM, once the switch is turned off or restarted, information is lost.
	Sysdb is very much like an event-driven publish/subscribe model. If the state of an agent changes, Sysdb will send an event notification to that agent, which will then update its local copy. Similarly when the agent writes to the mount point, it only changes its local copy and the write returns immediately.
	Each EOS agent subscribes to Sysdb to be notified when the states of other agents change in Sysdb. When the state of an agent has been changed, this change notification is buffered and asynchronously sent to Sysdb, which then notifies all other agents who have subscribed to the changed agent.
	This centralized database approach to passing state throughout the system, and the automated way the Sysdb code is generated reduces risk and error, improves software feature velocity, and provides flexibility for customers who can use the same APIs to receive notifications from Sysdb to customize and extend switch features.").
	See e.g., EOS: An Extensible Operating System (App. M, Ex. 15) at p. 3 ("Sysdb holds all state, while agents perform all processing. Sysdb is an address space that purely holds state and delivers state updates from one agent to another. For example, when a link goes down, a port driver updates the link state in Sysdb, which delivers the update to the SNMP service, which then sends a trap. Agents may contain copies of Sysdb state for efficiency, but all state is recoverable from Sysdb whenever needed. Like a traditional database engine, Sysdb itself is ultrareliable because it contains no application code.").
	See e.g., EOS: An Extensible Operating System (App. M, Ex. 15) at p. 3 ("Sysdb holds all state, while agents perform all processing. Sysdb is an address space that purely holds state and delivers state updates from one agent

Case 5:14-cv-05344-BLF Document 306-13 Filed 06/13/16 Page 34 of 34

U.S. 7,162,537	Arista Products
	to another. For example, when a link goes down, a port driver updates the link state in Sysdb, which delivers the update to the SNMP service, which then sends a trap. Agents may contain copies of Sysdb state for efficiency, but all state is recoverable from Sysdb whenever needed. Like a traditional database engine, Sysdb itself is ultrareliable because it contains no application code.").